UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/896,526	06/28/2001	Haitham Akkary	42390P11201	8421

8791 7590 03/11/2009

BLAKELY SOKOLOFF TAYLOR & ZAFMAN LLP
1279 OAKMEAD PARKWAY
SUNNYVALE, CA 94085-4040

EXAMINER	
HUISMAN, DAVID J	

ART UNIT	PAPER NUMBER
2183	

MAIL DATE	DELIVERY MODE
03/11/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	09/896,526	AKKARY ET AL.	
	Examiner	Art Unit	
	DAVID J. HUISMAN	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *07 January 2009*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-7,9-11,13,15-22 and 24-36* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-7,9,10,20-22 and 24-36* is/are rejected.

7)☒ Claim(s) *11,13 and 15-19* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *7/20/04, 12/20/04, & 3/25/05* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-7, 9-11, 13, 15-22, and 24-36 have been examined.

Papers Submitted

2. It is hereby acknowledged that the following papers have been received and placed of record in the file: RCE and Amendment as received on 1/7/2009.

Specification

3. The title of the invention is objected to because it is not clear what is meant by "control information commitment of results". Please clarify.

Claim Objections

4. Claim 5 is objected to because of the following informalities: Delete the space before "ordering". Appropriate correction is required.

5. Claim 6 is objected to because of the following informalities:

- In line 1, insert --wherein-- after the comma.
- In line 2, replace "comprising" with --comprises--.

Appropriate correction is required.

6. Claim 7 is objected to because of the following informalities:

- In line 1, insert --wherein-- after the comma.
- In line 1, replace "comprising" with --comprises--.

Appropriate correction is required.

7. Claim 10 is objected to because of the following informalities:

- In line 1, insert --wherein-- after the comma.
- In line 1, replace "comprising" with --comprises--.

Appropriate correction is required.

8. Claim 11 is objected to because of the following informalities: In line 6, is applicant trying to claim that a register is one of loaded from a register file buffer and written by said second processor? If so, please delete the comma after "buffer". If not, it is not clear what the other option is (implied by "one of"). Appropriate correction is required.

9. Claim 20 is objected to because of the following informalities:

- In line 4, replace "by" with --in--.
- In line 5, replace "from" with --in--.
- In line 8, is applicant trying to claim that a register is one of loaded from a register file buffer and written by said second processor? If so, please delete the comma after "buffer". If not, it is not clear what the other option is (implied by "one of"). Appropriate correction is required.

Appropriate correction is required.

10. Claim 30 is objected to because of the following informalities: Insert --wherein-- after the comma. Appropriate correction is required.

11. Claim 31 is objected to because of the following informalities: Insert --wherein-- after the comma. Appropriate correction is required.

12. Claim 33 is objected to because of the following informalities:

- In line 1, insert --wherein-- after the comma.

- In line 2, replace "sharing" with --share--.

Appropriate correction is required.

13. Claim 34 is objected to because of the following informalities: Insert a dash between "load" and "ordering" to be consistent with claim 5. Appropriate correction is required.

14. Claim 35 is objected to because of the following informalities:

- In line 1, insert --wherein-- after the comma.
- In line 1, replace "including" with --includes--.

Appropriate correction is required.

Claim Rejections - 35 USC § 101

15. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

16. Claims 20-22 and 24-28 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. Specifically, in claim 20, applicant claims a machine-readable storage medium containing instructions. Paragraph [0038] of the specification sets forth both statutory and non-statutory examples (signals) of machine-readable media. While applicant has deleted the language associated with propagation signals in the amendment filed on January 7, 2009, the deletion alone does not remove it from the original disclosure. Absent a clear disavowal, this amendment, which was not suggested by the examiner, does not overcome the 101 rejection. That is, absent a clear disavowal of claims to embodiments defined by the signals set forth in the original specification, it will be assumed that applicant still intends for the

medium of claims 28-30 to include transmission signals. Therefore, this rejection will be maintained until applicant provides a clear disavowal of the non-statutory subject matter.

Claim Rejections - 35 USC § 103

17. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

18. Claims 1-7, 9-10, and 29-36 are rejected under 35 U.S.C. 103(a) as being unpatentable over Nair et al., U.S. Patent No. 7,017,073 (herein referred to as Nair), in view of Hennessy and Patterson, "Computer Architecture - A Quantitative Approach, 2nd Edition," 1996 (herein referred to as Hennessy), and further in view of Sundaramoorthy et al., "Slipstream Processors: Improving both Performance and Fault Tolerance", 2000 (herein referred to as Sundaramoorthy), and the examiner's taking of Official Notice.

19. Referring to claim 1, Nair has taught an apparatus comprising:

a) a first processor (Fig.1, component 1) and a second processor (Fig.1, component 2) each having a decoder. Note that since each processor receives and executes its own instructions, as shown in Fig.1, each processor inherently has a decoder, as instructions must be decoded before they are executed.

b) a plurality of memory devices coupled to the first processor and the second processor. See Fig.1, component 14, and column 4, line 6, to column 5, line 18. At least a PROM and cache are coupled to the processors.

c) a second buffer coupled to the first processor and the second processor, the second buffer being a trace buffer. See Fig.1, at least component 12, which includes branch outcomes (trace information).

d) wherein the first processor and the second processor perform single threaded applications using multithreading resources. See Fig.2 and note that a single threaded application ***A*** is executed using multithreaded resources (i.e., simultaneous multithreaded (SMT) processors 0 and 1).

e) the first processor executes a single threaded application ahead of the second processor executing said single threaded application to avoid misprediction, and said single threaded application is not converted to an explicit multiple thread application. See column 4, lines 30-65, and note that thread ***A*** is executed ahead of ***A'***.

f) the single threaded application executed on the second processor avoids branch mispredictions from information received from said first processor. See column 4, lines 30-65, and Fig.1, and note that thread ***A*** is executed ahead of the other thread ***A'*** so that branch outcomes may be passed to ***A'*** to avoid misprediction.

g) Nair has not explicitly taught that the first and second processors each have a scoreboard. However, Hennessy has taught that a scoreboard allows instructions to execute out of order. As is known in the art, out-of-order execution is advantageous because it allows instructions to execute as soon as their resources are ready, thereby reducing stalling and CPU idleness. See Hennessy, pages 241 and 242. As a result, in order to allow both processors to benefit from such execution and resulting advantages, it would have been obvious to one of ordinary skill in the art

at the time of the invention to modify each of the first and second processors of Nair to include scoreboards.

h) Nair has not taught a first buffer coupled to the first processor and the second processor, the first buffer being a register buffer and is operable to transfer register values from the second processor to the first processor. However, Sundaramoorthy has taught the concept of passing register values, in addition to branch outcome values, between processors using a buffer (Fig.1, delay buffer) so that a trailing thread (R-stream) may utilize the values already computed by the leading thread (A-stream), thereby more efficiently executing the R-stream. See column 10, lines 17-21, and lines 35-38. As a result, in order to speed up execution of a trailing thread, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Nair to include a first buffer coupled to the first processor and the second processor, the first buffer being a register buffer and is operable to transfer register values from the second processor to the first processor. Based on the examiner's interpretation, Nair, as modified by Sundaramoorthy, would include four buffers between the processors. The four buffers would include buffers 12 and 13, as shown in Fig.1 of Nair, and two additional buffers for passing register information from each processor, much like buffers 12 and 13. The first buffer would be the register buffer that passes data in the same direction as buffer 13 (from thread ***B*** to ***B'***).

i) Nair has not taught a plurality of memory instruction buffers coupled to the first processor and the second processor. However, Official Notice is taken that load buffers, store buffers, and reorder buffers, and their advantages, are well known and accepted in the art. Specifically, load and store buffers buffer long latency memory operations so that the pipeline may continue to perform other work. Also, load and store buffers contribute to efficient out-of-order execution of

loads and stores. Reorder buffers, on the other hand, are inherent in all out-of-order systems, because although instructions may execute out of order, they must be retired in order. The reorder buffer ensures that instructions are retired in order. As described above, out-of-order execution is advantageous because it allows instructions to execute as soon as their resources are ready, thereby reducing stalling and CPU idleness. Consequently, to reduce stalling, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement at least load, store, and reorder buffers in the system of Nair.

20. Referring to claim 2, Nair, as modified, has taught an apparatus as described in claim 1. Nair has not explicitly taught that the memory devices comprise a plurality of cache devices. Instead, Nair has only taught a single L2 cache. However Official Notice is taken that L1 caches are well known and accepted in the art, especially in systems that already have an L2 cache. An L1 cache is faster than an L2 cache, thereby speeding up access to most recently accessed data. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to further modify Nair to include an L1 cache.

21. Referring to claim 3, Nair, as modified, has taught an apparatus as described in claim 1. Nair has not taught that the first processor is coupled to at least one of a plurality of zero level (L0) data cache devices and at least one of a plurality of L0 instruction cache devices, and the second processor is coupled to at least one of the plurality of L0 data cache devices and at least one of the plurality of L0 instruction cache devices. However, Sundaramoorthy has taught such a concept. See Fig.1 and note that each processor is connected to a separate data cache (D-Cache) and instruction (I-Cache) which can be considered as zero-level caches because they are directly connected to the execute cores). By having individual caches, bus contention would be

reduced between processors since they wouldn't be fighting for the same cache. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to further modify Nair such that the first processor is coupled to at least one of a plurality of zero level (L0) data cache devices and at least one of a plurality of L0 instruction cache devices, and the second processor is coupled to at least one of the plurality of L0 data cache devices and at least one of the plurality of L0 instruction cache devices.

22. Referring to claim 4, Nair, as modified, has taught an apparatus as described in claim 3, wherein each of the plurality of L0 data cache devices store exact copies of store instruction data. Although this is not mentioned explicitly, it is deemed inherent to the design because as each processor is executing the same thread, the data caches in each processor must contain exact copies of data. And, this data is store instruction data because data that is stored to main memory is also stored in a data cache.

23. Referring to claim 5, Nair, as modified, has taught an apparatus as described in claim 1, wherein the plurality of memory instruction buffers includes at least one store forwarding buffer and at least one load-ordering buffer (recall from the rejection of claim 1 that is would have been obvious to modify Nair to include a load buffer and a store buffer, which forwards stores to main memory).

24. Referring to claim 6, Nair, as modified, has taught an apparatus as described in claim 5. Although Nair, as modified, has not explicitly taught that the at least one store forwarding buffer comprises a structure having a plurality of entries, each of the plurality of entries having a tag portion, a validity portion, a data portion, a store instruction identification (ID) portion, and a thread ID portion, such fields are well known in the art and all relate to tracking a particular

instruction and identifying data associated with that instructions. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to further modify Nair to include these fields.

25. Referring to claim 7, Nair, as modified, has taught an apparatus as described in claim 6. Although Nair, as modified. does not mention that the at least one load ordering buffer comprises a structure having a plurality of entries, each of the plurality of entries having a tag portion, an entry validity portion, a load identification (ID) portion, and a load thread ID portion, such fields are well known in the art and all relate to tracking a particular instruction and identifying data associated with that instructions. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to further modify Nair to include these fields.

26. Referring to claim 9, Nair, as modified, has taught an apparatus as described in claim 1. Nair, despite teaching trace queues 12, 13 in Fig.1, does not disclose that the trace buffer is a circular buffer having an array with head and tail pointers, the head and tail pointers having a wrap-around bit. However, "Official Notice" is taken that it is well known and expected in the art to implement a FIFO queue as a circular buffer with head and tail pointers wherein head and tail pointers have a wrap-around bit. A circular buffer is useful to implement in hardware because only the head and tail pointers need to be incremented/decremented instead of actually physically shifting entries. A wrap around bit would also be needed to indicate whether the pointer has wrapped around the end of the queue. Therefore, it would been obvious to one of ordinary skill in the art at the time of the invention to have implemented the FIFO queue as a circular buffer with head and tail pointers, the head and tail pointers having a wrap around bit

because it is known that a FIFO queue can be implemented as a circular buffer and it is easier to build in hardware.

27. Referring to claim 10, Nair, as modified, has taught an apparatus as described in claim 1. Nair, as modified, has not explicitly taught that the register buffer comprising an integer register buffer and a predicate register buffer. However, Official Notice is taken that integer registers and predicate registers are well known and expected in the art. By implementing integer registers, the system will be able to load and store integer data and perform integer operations quickly. Furthermore, by implementing predicate registers, the system will be able to achieve conditional execution of instructions without conditional branch instructions. Consequently, to achieve such functionality, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Nair to include an integer register buffer and a predicate register buffer in the register buffer (delay buffer).

28. Referring to claims 29-32, claims 29-32 are rejected for the same reasons set forth in the rejections of claims 1-4, respectively. Any differences between the claims are inconsequential, as the differences are inherently present in the prior art.

29. Referring to claim 33, Nair, as modified, has taught a system as described in claim 31. Nair has further taught sharing an L2 cache (column 4, line 66, to column 5, line 6), but has not taught that the first processor and the second processor each share a first level (L1) cache. However, Official Notice is taken that it is well known and expected in the art that processors in a multi-processor environment share an L1 cache. Such a scheme allows for the simplification of cache coherency in that both processors would be able to access the same up-to-date cache as opposed to one of the processors accessing out-of-date information in its own cache. Therefore,

it would have been obvious to one of ordinary skill in the art at the time of the invention to have the first and second processors share L1 cache.

30. Referring to claims 34-35, claims 34-35 are rejected for the same reasons set forth in the rejections of claims 5-6, respectively.

31. Referring to claim 36, Nair, as modified, has taught the system of claim 29. Although Nair has not explicitly taught that the second processor is operable to commit results in one commit cycle based at least on the information received from the first processor, the examiner asserts that pipelining is well known and advantageous in the art. Specifically, pipelining is the breaking up of tasks into stages, where each stage processes a different instruction at the same time, thereby increasing parallelism and throughput. A common pipeline includes fetch, decode, execution, and writeback stages. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Nair such that the processors are pipelined. It then follows that writing of results occurs in one cycle (in the writeback stage, where results are written to storage). And since, the second processor's execution is helped by information received by the first processor, it can be said that the second processor is operable to commit results in one commit cycle based at least on the information received from the first processor

32. Claims 20-22 and 24-28 are rejected under 35 U.S.C. 103(a) as being unpatentable over Nair in view of Hennessy and further in view of Sundaramoorthy, Akkary, WO 99/31594, and Tanenbaum, "Structured Computer Organization", 1984, (herein referred to as Tanenbaum).

33. Referring to claim 20, Nair has taught:

a) executing a single thread by a first processor. See Fig.1, component 2, which executes thread ***A'***.

b) executing said single thread from a second processor (see Fig.1, component 1, which executes thread ***A***, thread ***A*** being the same as thread ***A'***, only behind in execution) as directed by the first processor, the second processor executing instructions ahead of the first processor to avoid misprediction. See column 4, lines 30-65.

c) wherein the first processor and the second processor execute single threaded applications using multithreading resources, said single thread is not converted to an explicit multiple-thread application, and said single thread contains the same number of instructions when executed on said first processor and said second processor. See Fig.2 and note that a single thread ***A*** is executed using multithreaded resources (i.e., simultaneous multithreaded (SMT) processors 0 and 1). Also, ***A*** and ***A'*** are the same threads, and therefore, have the same number of instructions. There is no conversion to multiple threads.

d) said single thread executed on the second processor avoids branch mispredictions from information received from said first processor. See column 4, lines 30-65, and Fig.1, and note that thread ***A*** is executed ahead of the other thread ***A'*** so that branch outcomes may be passed to ***A'*** to avoid misprediction.

e) Nair has not taught tracking at least one register that is one of loaded from a first buffer, and written by said second processor, said tracking executed by said second processor, the first buffer being a register file buffer. However, Hennessy has taught the idea of a scoreboard which allows instructions to execute out of order. As is known in the art, out-of-order execution is advantageous because it allows instructions to execute as soon as their resources are ready,

thereby reducing stalling and CPU idleness. See pages 241 and 242. As a result, in order to allow the second processor to benefit from such execution and resulting advantages, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the second processor of Nair to include a scoreboard. And, the inherent nature of a scoreboard is to track registers written by the second processor. See Fig.4.4 on page 247 of Hennessy, and note that the system tracks when registers are ready so that execution may continue. For registers to be ready, it must be tracked when the writing to those registers completes.

f) Nair has not taught clearing a store validity bit and setting a mispredicted bit in a load entry in a second buffer if a replayed store instruction has a matching store identification (ID) portion, the second buffer being a trace buffer. However, Official Notice is taken that it is well known and expected in the art to use load and store buffers for the proper handling of memory operations. Akkary discloses a system for ordering loads and stores in a multithreaded processor using load and store buffers (fig. 2, 182,184). He discloses clearing a store validity bit (SB Hit field) in the load buffer if data came from memory (pg. 37, para. 3, line 4; pg. 38, line 1). Also when a store instruction is executed (which includes replayed stores), its address is compared with the store ID portion (addresses) of load instructions (pg. 36, para. 3). On a match, a replay event is signaled to the load entry in the trace buffer to replay the load instruction and all its dependant instructions because it was mispredicted (pg. 38, para. 2). Furthermore, Official Notice is taken that is well known and expected in the art to set a status bit to indicate a misprediction. Clearly, in order to detect a misprediction, some bit must change somewhere in the system. As shown in In re Larson, 144 USPQ 347 (CCPA 1965), to make integral is generally not given patentable weight or would have been an obvious improvement. That is, it does not matter where this

misprediction bit is located within the system, as long as it exists. One of ordinary skill in the art would have recognized that one could use the load and store buffer arrangement of Akkary in the Nair reference in order handle loads and stores in the multithreaded environment. Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Nair by clearing a store validity bit and setting a mispredicted bit in a load entry in a second buffer if a replayed store instruction has a matching store ID portion.

g) Nair has not taught an apparatus comprising a machine-readable medium containing instructions which, when executed by a machine to perform the aforementioned operations. However, Tanenbaum has taught that any instruction executed by hardware can also be simulated in software (pg 11, para. 4, lines 1-2). He also teaches that hardware is generally immutable (first para. after sec. 1.4 header) while software allows for more rapid change (pg. 11, para. 4, lines 2-4). One of ordinary skill in the art at the time of the invention would have been motivated to convert the Nair reference to software i.e. instructions on a machine readable medium because Tanenbaum teaches that hardware is generally immutable (first para. after sec. 1.4 header) while software allows for more rapid change (pg. 11, para. 4, lines 2-4). Therefore, to allow for ease of correction of mistakes, and/or an ease of addition of new functionality, it would have been obvious to one of ordinary skill in the art to have implemented the method of Nair by an apparatus comprising instructions recorded on a machine readable medium.

34. Referring to claim 21, Nair, as modified, has taught an apparatus as described in claim 20. Nair has further taught transmitting control flow information from the second processor to the first processor, the first processor avoiding branch prediction by receiving the control flow information. See Fig.1, and column 4, lines 56-65.

35. Referring to claim 22, Nair, as modified, has taught an apparatus as described in claim 21. Nair has not taught duplicating memory information in separate memory devices for independent access by the first processor and the second processor. However, implementing a cache for each processor is well known and accepted in the art. A cache allows for fast access to recently accessed data and by having a separate cache for each processor, bus contention is reduced since both processors are not fighting each other for use of the cache. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Nair to include separate memory devices for each processor. And, because each processor executes identical threads, the contents of the caches would inherently be equivalent (i.e., duplicated).

36. Referring to claim 24, Nair, as modified, has taught a method as described in claim 21. Nair has not taught setting a store validity bit if a store instruction that is not replayed matches a store identification (ID) portion. However, Official Notice is taken that it is well known and expected in the art to use load and store buffers for the proper handling of memory operations. Akkary discloses a system for ordering loads and stores in a multithreaded processor using load and store buffers (fig. 2, 182,184). He discloses setting a store validity bit (SB Hit field) in the load buffer if data came from store buffer (pg. 37, para. 3, line 4; pg. 38, lines 1-2). In order for data to come from the store buffer, a store instruction address (including store instructions that are not replayed) must match a store ID portion (address) of the load entry. One of ordinary skill in the art would have recognized that one could use the load and store buffer arrangement of Akkary in the Nair reference in order handle loads and stores in the multithreaded environment. Therefore it would have been obvious to one of ordinary skill in the art at the time of the

invention to have modified the Nair reference by setting a store validity bit if a store instruction that is not replayed matches the store ID portion.

37. Referring to claim 25, Nair, as modified, has taught a method as described in claim 21. Furthermore, Nair has not taught flushing a pipeline, setting a mispredicted bit in a load entry in the second buffer and restarting a load instruction if one of the load is not replayed and does not match a tag portion in a load buffer, and the load instruction matches the tag portion in the load buffer while a store valid bit is not set. However, Official Notice is taken that it is well known and expected in the art to use load and store buffers for the proper handling of memory operations. Akkary discloses a system for ordering loads and stores in a multithreaded processor using load and store buffers (fig. 2, 182,184). In particular, when a store valid bit is not set (SB hit = 0, pg. 38, para. 2) and when a store instruction compared with the addresses of load instructions (pg. 36, para. 3) is a match, a replay event is signaled to the load entry in the trace buffer to replay the load instruction and all its dependant instructions because it was mispredicted (pg. 38, para. 2). Furthermore, Official Notice is taken that is well known and expected in the art to set a status bit to indicate a misprediction. Clearly, in order to detect a misprediction, some bit must change somewhere in the system. As shown in In re Larson, 144 USPQ 347 (CCPA 1965), to make integral is generally not given patentable weight or would have been an obvious improvement. That is, it does not matter where this misprediction bit is located within the system, as long as it exists. One of ordinary skill in the art would have recognized that one could use the load and store buffer arrangement of Akkary in the Nair reference in order handle loads and stores in the multithreaded environment and flush the pipeline on reading the mispredicted bit. Therefore it would have been obvious to one ordinary

skill in the art at the time of the invention to have modified the Nair reference by flushing a pipeline, setting a mispredicted bit in a load entry in the trace buffer and restarting a load instruction if one of the load is not replayed and does not match a tag portion in a load buffer, and the load instruction matches the tag portion in the load buffer while a store valid bit is not set.

38. Referring to claim 26, Nair, as modified, has taught an apparatus as described in claim 21. Nair has further taught:

a) executing a replay mode at a first instruction of a speculative thread. This is deemed inherent because when the check thread ***A'*** is initially started, i.e., at the first instruction, there will be two redundant threads being executed which means the thread is being replayed from the beginning. This can be called a replay mode. Also, any thread which predicts branches is a speculative thread. Hence, the threads of Nair are speculative.

b) terminating the replay mode and the execution of the speculative thread if a partition in the second buffer is approaching an empty state. This limitation is also deemed inherent to the reference because when the partition in the buffer is approaching an empty state that means that thread ***A*** has stopped producing results and finished executing. Therefore, at that point the replay mode and thread ***A*** are terminated).

39. Referring to claim 27, Nair, as modified, has taught a method as described in claim 21. Nair has further taught:

a) issuing all instructions up to a next replayed instruction including dependent instructions (This feature is deemed inherent to the design because in order to execute the thread all instructions are issued in either one of the processors).

b) issuing instructions that are not replayed as no-operation (NOPs) instructions (This feature is also deemed inherent to the design because if an instruction that is not replayed does not occupy a slot in the execution pipeline it will lead to improper functioning of the processor. Hence as the instruction that is not replayed is not to be executed, a NOP must be issued in its place).

c) issuing all load instructions and store instructions to memory (This limitation is also deemed inherent to the design because if all loads and stores are not issued to memory, the state of the thread would be incorrect leading to the malfunctioning of the system).

d) committing non-replayed instructions from the second buffer to a register file (Although this is not explicitly mentioned, it is deemed inherent to the design because a register file exists within the processors and as results are written to the register file so that they can be read from by future instructions, the results of the instructions from the buffer that are not going to be replayed must be written into the register file).

e) Nair has not taught supplying names from the second buffer to preclude register renaming. However, Hennessy has taught that register renaming is used to reduce name dependencies allowing instructions involved in name dependencies to execute simultaneously or be reordered (pg. 232, para. 5). As these dependencies are resolved, more instruction level parallelism can be extracted and performance can be improved. One of ordinary skill in the art would have recognized to use register renaming in the Nair reference because it too would improve performance. As the buffer would also supply the names, it would be logical not to do the renaming again in the first processor. Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to have modified the Nair reference by adding register renaming capabilities and supply names from the trace buffer to preclude register renaming. One

would have been motivated to do so because it would improve performance which is an objective of every processor design.

40. Referring to claim 28, Nair, as modified, has taught an apparatus as described in claim 21. Nair has not taught clearing a valid bit in an entry in a load buffer. However, a reorder buffer is implemented in out-of-order systems, because although instructions may execute out of order, they must be retired in order. The reorder buffer ensures that instructions are retired in order. As is known, out-of-order execution is advantageous because it allows instructions to execute as soon as their resources are ready, thereby reducing stalling and CPU idleness. Consequently, to reduce stalling, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement a reorder buffers (i.e., load buffer) in the system of Nair. And, although not explicitly mentioned, clearing a validity bit is deemed inherent to the design because a load entry, on being retired, has to be marked invalid to ensure that other new instructions can occupy that entry safely.

Allowable Subject Matter

41. Claims 11, 13, and 15-19 are allowed. The examiner asserts that after reconsideration, the prior art of record has not taught, nor rendered obvious, the transmitting results from the second processor to the first processor, the first processor avoiding executing a portion of instructions by committing the results of the portion of instruction into a register file from a first buffer, the first buffer being a trace buffer.

Please correct any objections associated with these claims.

Response to Arguments

42. Applicant's arguments with respect to the prior art as applied to the claims have been considered but are moot in view of the new ground(s) of rejection.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to DAVID J. HUISMAN whose telephone number is (571)272-4168. The examiner can normally be reached on Monday-Friday (8:00-4:30).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/David J. Huisman/
Primary Examiner, Art Unit 2183
February 12, 2009